Claims

[c1] A method for managing interrupts in a multiple virtual machine environment, comprising the steps of:
running concurrently a plurality of independent virtual machines, each virtual machine having associated therewith a plurality of anticipated interrupt signal types;
receiving a plurality of interrupt signals;
determining which interrupt signal of the plurality of received interrupt signals has the highest priority; and
servicing the interrupt signal determined to have the highest priority.

[c2] The method of claim 1, wherein said running step comprises running at least two Java virtual machines.

[c3] The method of claim 3, further comprising the step of activating a specific independent virtual machine of said plurality of independent virtual machines.

[c4] The method of claim 3, wherein said activating step comprises the step of using a timer to define an activation period of an activated virtual machine.

[c5] The method of claim 3, further comprising the step of assigning a memory region to at least one independent virtual machine of the plurality of independent virtual machines.

[c6] The method of claim 5, further comprising the step of protecting a virtual machine's memory region from accesses by a different virtual machine.

[c7] The method of claim 6, wherein said protecting step comprises the steps of:
screening a memory access; and
generating an abort interrupt signal to abort an access to a memory region of a nonactivated virtual machine.

[c8] The method of claim 5, further comprising the step of outputting the identity of the activated virtual machine to a memory management component.

[c9]

The method of claim 8, further comprising the step of identifying, by the memory management component, the memory region assigned to the activated virtual machine.

[c10] The method of claim 9, further comprising the step of monitoring address lines to abort attempted memory accesses to a protected memory region.

[c11] The method of claim 10, further comprising the step of aborting an attempted access to a protected memory region by generating an error signal.

[c12] The method of claim 10, further comprising the step of aborting an attempted access to a protected memory region by generating a prioritized nonmaskable interrupt signal.

[c13] The method of claim 10, further comprising the step of aborting an attempted access to a protected memory region by generating a highest priority prioritized nonmaskable interrupt signal.

[c14] The method of claim 3, wherein said receiving step comprises receiving a maskable interrupt signal.

[c15] The method of claim 14, further comprising the step of latching a received maskable interrupt signal.

[c16] The method of claim 14, further comprising the step of latching a received maskable interrupt signal into a virtual interrupt latch component even though the independent virtual machine with which it is associated is not the activated independent virtual machine at the time the received maskable interrupt signal is received.

[c17] The method of claim 16, further comprising the step of transferring the maskable interrupt signal, upon activation of its associated virtual machine, from the virtual interrupt latch component to a global interrupt mask register.

[c18] The method of claim 17, further comprising the step of transferring the maskable interrupt signal, upon activation of its associated virtual machine, from the virtual interrupt latch component to a local mask register.

[c19] The method of claim 18, further comprising the step of sending the maskable

interrupt signal, upon activation of its associated virtual machine, to a priority encoder after said steps of transferring and communicating.

[c20] The method of claim 16, further comprising the steps of:
holding the received maskable interrupt signal in the virtual interrupt latch component until the independent virtual machine with which it is associated has been activated; and
servicing the received maskable interrupt signal during the time period that its associated independent virtual machine has been activated.

[c21] The method of claim 14, further comprising the steps of:
discerning whether the independent virtual machine associated with the received maskable interrupt signal is the activated independent virtual machine; and
ignoring the received maskable interrupt signal if it is discerned that the independent virtual machine with which the received maskable interrupt signal is associated is not the currently activated independent virtual machine.

[c22] The method of claim 1, wherein said receiving step comprises receiving a nonmaskable interrupt signal.

[c23] The method of claim 1, wherein said receiving step comprises receiving a nonmaskable interrupt signal indicating a power supply interruption.

[c24] The method of claim 3, wherein said receiving step comprises receiving a nonmaskable interrupt signal indicating activation of a different independent virtual machine.

[c25] The method of claim 1, wherein said receiving step comprises receiving a nonmaskable interrupt signal indicating an application specific event.

[c26] The method of claim 1, wherein said receiving step comprises receiving a nonmaskable interrupt signal indicating a prohibited memory access attempt.

[c27] The method of claim 3, further comprising the step of reserving the highest priority for interrupt signals indicating a prohibited memory access attempt; and wherein said receiving step comprises receiving a nonmaskable interrupt

signal indicating a prohibited memory access attempt .

[c28] The method of claim 27, further comprising the step of suspending execution of the activated independent virtual machine upon receipt of a nonmaskable interrupt signal indicating a prohibited memory access attempt.

[c29] An interrupt management system for an apparatus capable of running multiple concurrent virtual machines, comprising:
a timer component comprising a plurality of virtual machine timers, said timer component further comprising an active virtual machine switch signal output;
a multiple virtual machine control component, comprising an active virtual machine identification signal output;
a processor component, coupled with said timer component;
an interrupt controller component coupled with said processor component and with said timer component, said interrupt controller component comprising an active virtual machine identification signal input coupled with said active virtual machine identification signal output, said interrupt controller component also comprising an interrupt signal input; and
a memory component storing interrupt handler code.

[c30] The interrupt management system of claim 29, wherein said interrupt controller component further comprises a plurality of virtual interrupt latch components.

[c31] The interrupt management system of claim 30, further comprising a plurality of global interrupt mask registers.

[c32] The interrupt management system of claim 30, further comprising a plurality of global interrupt mask registers, and wherein each global interrupt mask register is coupled with one of the virtual interrupt latch components.

[c33] The interrupt management system of claim 32, further comprising a local mask register coupled with said plurality of global interrupt mask registers.

[c34] The interrupt management system of claim 33, further comprising a priority encoder coupled with said local mask register.

[c35] An interrupt controller for a multiple virtual machine environment, comprising:<

an interrupt signal input;
a plurality of virtual interrupt latch components coupled with said interrupt signal input; and
a plurality of global interrupt mask registers;
wherein each global interrupt mask register of said plurality of global interrupt mask registers is coupled with one of the virtual interrupt latch components.

[c36] The interrupt controller of claim 35, further comprising a local mask register coupled with said plurality of global interrupt mask registers.

[c37] The interrupt controller of claim 36, further comprising a priority encoder coupled with said local mask register.

[c38]

A processor-based interrupt signal management system for a multiple virtual machine environment, comprising:
an integrated circuit chip, comprising;
a processor component;
a multiple virtual machine management component coupled with said processor component, said multiple virtual machine management component comprising a plurality of virtual machine activation timer components;
a memory component coupled with said processor component, said memory component comprising interrupt handler code;
a memory access error input;
an active virtual machine identification output; and
a memory access location output; and
an external memory protection component, not located on said integrated circuit chip, comprising an active virtual machine identification input and a memory access location input, said active virtual machine identification input coupled with said active virtual machine identification output and said memory access location input coupled with said memory access location output of said integrated circuit chip, said external memory protection component comprising a memory access error output, said memory access error output coupled with said memory access error input;
wherein said external memory protection component indicates a memory access

error via said memory access error output when said memory access location input indicates memory location not associated with a virtual machine identified by said active virtual machine identification output.